ArThUR: A Tool for Markov Logic Network

Bodart, Axel; Evrard, Keyvin; Ortiz Vega, James Jerson; Schobbens, Pierre Yves

Published in:
Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)

Publication date:
2014

Document Version
Early version, also known as pre-print

ArThUR: A Tool for Markov Logic Network

Axel Bodart, Keyvin Evrard, James Ortiz, and Pierre-Yves Schobbens

Computer Science Faculty, University of Namur
{axbodart,kevrard}@student.fundp.ac.be, {jor,pys}@info.fundp.ac.be

Abstract. Logical approaches-and ontologies in particular-offer a well-adapted framework for representing knowledge present on the Semantic Web (SW). These ontologies are formulated in Web Ontology Language (OWL2), which are based on expressive Description Logics (DL). DL are a subset of First-Order Logic (FOL) that provides decidable reasoning. Based on DL, it is possible to rely on inference mechanisms to obtain new knowledge from axioms, rules and facts specified in the ontologies. However, these classical inference mechanisms do not deal with : uncertainty probabilities. Several works recently targeted those issues (i.e. Pronto, PR-OWL, BayesOWL, etc.), but none of them combines OWL2 with Markov Logic Networks (MLN) formalism. Several open source software packages for MLN are available (e.g. Alchemy, Tuffy, RockIt, etc.). In this paper, we present ArThUR, a Java framework for reasoning with probabilistic information in the SW. ArThUR incorporate three open source software packages for MLN, which is able to reason with uncertainty information, showing that it can be used in several real-world domains. We also show several experiments of our tool with different ontologies.

1 Introduction

In the last years, the Semantic Web (SW) [2] has acquired significant relevance in academic, industrial, military, health care and biological. SW is currently the major commitment of the W3C[1] and to offer the greatest evolution of the World Wide Web (Web). The main ideas behind the SW are : the implementation of cooperative agents and methods for processing the information in the Web, the uses of ontologies [7] for a precise definition of shared terms in the Web and using reasoning languages for automated inference from ontologies [2]. Ontologies are common vocabularies that define concepts and their relationships. The standard Web Ontology Language is OWL2 [14] from the W3C. OWL2 is an expressive Description Logic (DL) ($\mathcal{SROIQ}(\mathcal{D})$) [1] with an RDF syntax [9]. More generally, DL is a type of logic to describe concepts and relationships. It is based on First-Order Logic (FOL) with the idea to formalize SW. Moreover, DL allows the use of deterministic reasoner tools (i.e. Pellet[2], FaCT++ [3], TrOWL[4], etc.) and verifying

[1] http://wwww.W3C.org
[2] http://wwww.mindswap.org/pellet
[3] http://owl.cs.manchester.ac.uk/tools/fact/
[4] http://trowl.eu/

R. Meersman et al. (Eds.): OTM 2014 Workshops, LNCS 8842, pp. 319–328, 2014.

if statements are true or false. However, this DL is less effective in those areas where the information to be represented comes along with uncertainty. For example, how to represent the uncertain relations between bacteria and antibiotics in statements such as: "Mycobacterium Smegmatis is resistant to Rifampicin with the degree of certainty 90% and is not resistant with he degree of certainty 10%". Fortunately, DL was extended to deal with uncertainty, especially was extended the logic behind DL in order to be able work with the uncertainty representations and reasoners. Several formalisms and tools for dealing with the knowledge uncertainty SW have been successfully applied over the last years. Formalisms like : Bayesian Networks [8], Fuzzy Logic [11], Markov Logic Networks (MLN)[6]. Probabilistic Description Logic (PDL) like : P-$\mathcal{SHOIN}(\mathcal{D})$ [10], P-$\mathcal{SHIF}(\mathcal{D})$ [13]. Reasoner tools like : Pronto[5], Incerto[6], ContraBovemRufum [15] and Probabilistic OWL like PR-OWL[7], BayesOWL [5], OntoBayes [19].

Several open source software packages for MLN (reasoners MLN) are available in the Web like Alchemy[8], Tuffy[9], ProbCog[10], and Markov TheBeast[11], RockIt [12] but none of them combines their MLN formalism with OWL2. Furthermore, these implementations do not present a very friendly graphical user interface that would facilitate the creating, editing and making probabilistic inferences from any ontology. Also in some of them there is only command line interface.

In this paper is presented ArThUR, an implementation in Java that consists of a graphical user interface and which combines three reasoners MLN (Alchemy, Tuffy, Rockit). The aims of ArThUR are to ease the modeling, editing and translating from OWL2 ontologies to MLN files. Furthermore, ArThUR makes probabilistic inferences, comparison, research and statistical result analysis of the probabilistic rules present in the MLN files. Finally, it is also possible to edit and to persist these structures as a standard MLN file and OWL2 ontology.

Structure of the paper. The rest of the paper is organized as follows. In sections 2, we introduce the ontologies and SW. In section 3, we give a brief introduction to MLN, uncertainty in the SW. In section 4, we present some open source software packages for MLN. In section 5, we present the ArThUR architecture. In section 6, we present the experimental work done and its main results. Finally, in section 7, we conclude and outline future work.

2 Background

2.1 Semantic Web

The SW is a project of the evolution of our current Web (Web 2.0) initiated by the W3C. This initiative is also known under the names of : Linked Data, Linked

[5] http://weblog.clarkparsia.com/2007/09/27/introducing-pronto
[6] https://code.google.com/p/incerto/
[7] http://www.pr-owl.org/
[8] http://alchemy.cs.washington.edu/
[9] http://hazy.cs.wisc.edu/hazy/tuffy/
[10] http://ias.in.tum.de/research/probcog
[11] https://code.google.com/p/thebeast/
[12] https://code.google.com/p/rockit/

Open Data or Linking Open Data. The main aim of this movement is to ensure that the amount of data displayed on the Web is also available in a standard format, accessible and manipulable in an unified manner by all Web applications.

2.2 Ontologies and Inferences

The term ontology is strongly linked to the SW. Several definitions have been given, including [7] and [16]. The first gives a very abstract definition of ontologies though popular in the community : "An ontology is an explicit specification of a conceptualization." The second proposes the following: "An ontology defines the basic terms and relations comprising the vocabulary of a topic area as well as the rules for combining terms and relations to define extensions to the vocabulary". Ontologies are used to capture knowledge about some domain of interest.

Inference is the process to deriving new knowledge from existing facts. In the case of ontologies, it is practical to discover new triplets from known triplets by exploiting the structure of the ontology and the logical constraints specified. Inference highlights implicit facts that are otherwise hidden in complex models. These inference mechanisms can be used to check the quality of knowledge bases and to add implicit knowledge to the ontologies. Various reasoners support deterministic inference (e.g. Pellet, FaCT++, HermiT, etc).

3 Logic and Uncertainty

3.1 Uncertainty and Semantic Web

Uncertainty is a fundamental and inevitable property present in many domains, including the Medical Domain (MD). Uncertainty in MD is represented with probabilistic approaches (e.g. represent the uncertain relations between Breast Cancer (BRC) and European woman in statements such as "European women have 12% risk of developing BRC in their lifetime"), which are considered as the scientific norm for this purpose. The majority of probabilistic approaches are based on probabilistic formalisms, such as Bayesian Networks [8] and Markov Logic (ML) [6].

There are several approaches combining probabilistic information and SW. A first approach is providing an ontology model to classify the probabilistic information (such as PR-OWL). A second approach to extend the ontology with primitives to represent the probabilistic information. For example: Pronto which extends the traditional DL's by associating a probability value to the axioms. And finally OWL2, FOL and probabilistic graphical models are currently being combined.

3.2 Markov Logic Networks

ML is a same language as FOL that provides that capability, joining in the same representation probabilistic graphical models to represent the uncertainty, and

FOL to represent a set of constraints on possible worlds. The main idea behind ML is that, unlike FOL, a world that violates a formula is not invalid, but less probable. This is done by attaching weights to FOL formulas: the higher the weight, the bigger is the difference between a world that satisfies the formula and one that does not, other things been equal. These sets of weighted formulas are called a ML network (MLN). Below, we consider two relevant tasks for MLN : Weight Learning and Inference [3][18].

4 Markov Logic Reasoning

A Markov Logic Reasoner (MLR) is composed of a set of algorithms that allows (weight) Learning and Inference based on MLN. Learning leads to estimating the weight associated with each formula of a given MLN [3]. Inference find the most likely state of the world consistent with some evidence, and computing arbitrary marginal or conditional probabilities [18]. The main MLN currently available are:

1. **Alchemy** was designed for Linux platforms. Alchemy is not portable due to its Linux C/C++ implementation, but nevertheless is currently the MLR most complete and most used. It includes implementations for all the major existing algorithms for structure learning, generative weight learning, discriminative weight learning, and inference [12]. The advantage of Alchemy it has a large community. The disadvantages are : (1) The grounding[13] takes very long and even for medium sized models the execution time might be very large [12]. (2) Hard formulas in Alchemy are not guaranteed to hold in the final MAP (Maximun A-Posteriori Query[14]) state. This is due to the fact that Alchemy only sets very large weights for hard formulas which may be violated in the MAP state [12].
2. **Tuffy** is implemented in Java programming language. Contrary to Alchemy, Tuffy does not offering different implementations of algorithms for learning and inference, but it offers those their have been proven to be among the most efficient experimentally. The Tuffy system is faster in performance than the Alchemy. The main disadvantage of Tuffy is the handling of negative weights in formulas. Tuffy proposes a bottom-up approach in its algorithms and combining the main memory and the use of a database to optimize processes. Additionally, Tuffy proposes partitioning techniques allowing networks to implement a parallelism strategy which significantly improves performance.
3. **Markov TheBeast (MTB)** is an open-source Statistical Relational Learning software based on MLN. It is developed in Java, it does not have much documentation. In contrast to Alchemy, MTB uses a different MAP inference technique : Integer Linear Programming with Cutting Planes.
4. **ProbCog** is an open source software for Statistical Relational Learning that supports learning and inference for relational domains. ProbCog is built on

[13] A ground term of a formal system is a term that does not contain any free variables.
[14] Informally, the MAP can be used to obtain a point estimate of an unobserved quantity on the basis of empirical data.

basis of PyMLN[15], a toolbox and a software library for learning and reasoning in MLN. Most of the code of ProgCog is developed in Java, although PyMLN is developed in Python.

5. **RockIt** is more recent and is also developed in Java programming language. A disadvantage here is that by the fact that it is relatively new, it is not yet entirely stable and changes often. In addition, it is very difficult to find documentation about its implementation. RockIt also uses a database, enables the parallelism (and implements optional techniques to improve performance).

4.1 Comparison of Reasoners

We have used several criteria for guide our choice of MLN, which are:

1. Methodology : This criteria indicates the use of database by the reasoners.
2. Platforms : This criteria indicates operating systems on which reasoners can work. i.e: Windows, Linux and Mac.
3. Runtime : This criteria indicates the time taken to perform queries for the set of concepts in the ontologies by the reasoners.
4. MLN Tasks : This criteria indicates the use of inferences and learning tasks by the reasoners.
5. Availability : This criteria ensures that the reasoners are freely available as open source.
6. Jena Support : This criteria indicates whether the reasoners can be used with Jena API[16] or not.
7. Friendly Grammar : This criteria indicates whether the grammar of the reasoners facilitates the extension and implementation phases.

	Alchemy	Tuffy	RockIt	MTB	ProbCog
Methodology	-	+	+	-	-
Platforms	+	+	+	+	+
Runtime	-	+	+	-	-
MLN Tasks	+	+	-	+	+
Availability	+	+	+	+	+
Jena Support	+	+	+	+	+
Friendly Grammar	+	+	+	-	-

5 Development

5.1 Architecture

ArThUR is using the three MLR (Alchemy, Tuffy, Rockit). It supports translating a OWL2 ontology into a MLN file, running learning and inference processes over the ontologies and statistical analysis of the results obtained. ArThUR has been written in Java and more specifically in Java SE 1.6 edition, an object-oriented

[15] http://alchemy.cs.washington.edu/
[16] https://jena.apache.org/

language known to provide platform independence and to enable fast and efficient program development. Furthermore, support for the development of Graphical User Interface (GUI) as well as grammar parsers are available. The design and implementation of ArThUR took approximately 5 months, with about 25000 lines of code for the kernel and 3000 lines of code for the GUI implementation. The tool architecture consists of five components, see Figure 1.

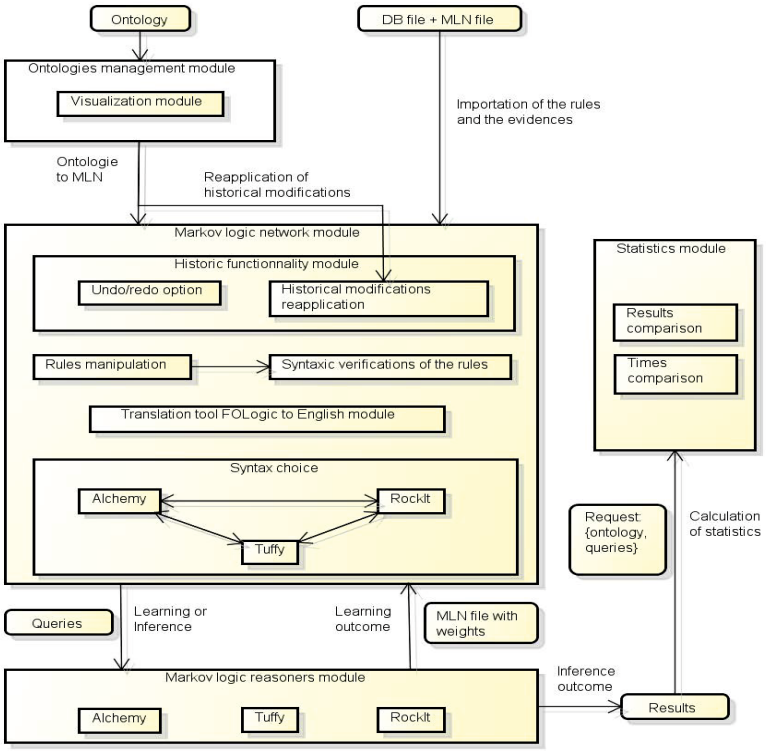

Fig. 1. The ArThUR architecture

1. GUI Component: In Figure 2, we can see the different options offered to the user. The user can load, save and visualize ontologies files. Furthermore, the user can translate ontologies files to MLN files, verify MLN rules and learn/infer from the MLN files using the three reasoners. The GUI prints results on screen or writes them into a file and allows the user to analyze the results and compare them with results achieved with other algorithms present in the reasoners.
2. Input Files Component: The input files are : a MLN file, an evidence file and a query file. The last one can be replaced allowing to user to choose what query predicates to use. Figure 2 shows how to load these three files and the possibility to select the query predicates through a choice box field.
3. FOL Rules Component: As the MLN file and the evidence files are being loaded, their terms (i.e. declarations, query, weighted formulas, rules (FOL)

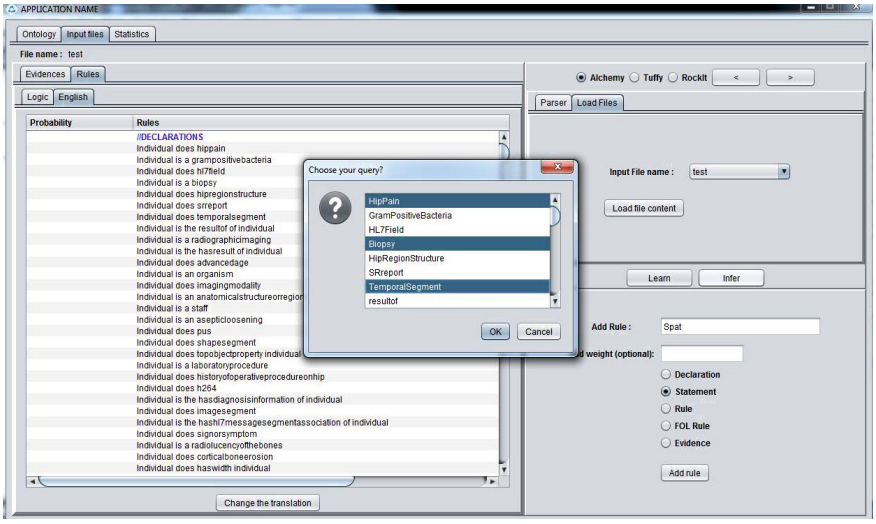

Fig. 2. User Interface of ArThUR

and evidences) are separated and organized in data structure in order to check the syntax in each term. Moreover, adding and removing terms are almost similar except that it necessary check that this terms does not already exist in the set of existing terms. Every change made through this features is persisted in the original file. This feature makes it easier for the user to include or remove terms in a MLN file. Furthermore, for the transformation of the FOL in Conjunctive Normal Form (CNF): (1) Eliminate conditional and bi-conditional of the logical formula. (2) Move negation operators inside of the logic formula. (3) Move the quantifies operators outside of the logical formula. (4) Apply skolemization. (5) Remove universal quantifiers operators. (6) AND to OR operators are distributed across the parentheses. An another processing step of this component is to translate FOL to natural language rules (English rules). We reuse our data structure to create transformation rules on the logic formula to english natural rules. The translation data is stored as java object and then stored in a file to ensure the persistence of these data.

4. Markov Logic Component: This is the component that supports inference and learning algorithms of the three reasoners. The weight learning consists in finding formula weights that maximize either the likelihood or pseudo-likelihood measure [3] for generative learning or either the conditional likelihood or max-margin measures [18] for discriminative learning. Inference consists in finding the marginal and conditional probabilities of a formula given an MLN, based on randomized and MC-SAT [17] algorithms.
5. Statistical Component: Finally, we developed a statistics component and comparison tools through an XML file, allowing us to share data persistence when closing the application and, secondly, giving us a system with a simple

structure that allows search and make specific requests. The XML structure is divided according to the different ontologies and according to the execution time and the results obtained for the three reasoners.

6 Experiments

In this section, we report out experiences with ArThUR in the context of MLN to learn and infer about uncertainty in SW. We ran the experiments on a 2.2GHz Intel core i7-2670 QM with 8 GO memory under the Windows 7 64 bits operating system. The main objective of these experiments is to show the efficiency and accuracy of the three MLR reasoners to learn and reason from an ontology, depending on the characteristics of the ontology which contains the knowledge base. To do this, we tested the different reasoners via ArThUR in order to achieve results and determine the most effective reasoners.

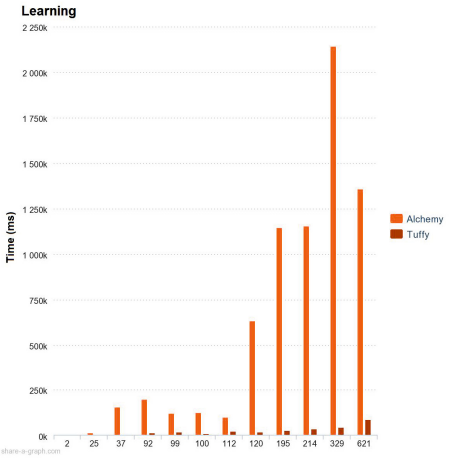

Fig. 3. Weight Learning Process

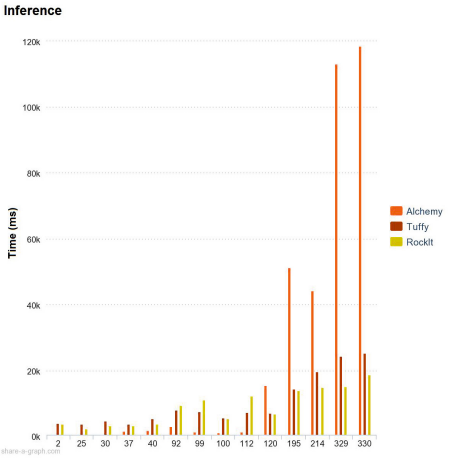

Fig. 4. Inference Process

Regarding MLR differences, we had to decide which algorithms to use for the different reasoners in the steps of learning and inference. For the Weight Learning Process, we have chosen the "Scaled Conjugate Gradient" algorithm for Alchemy because it appears that it is the most effective method in [18]. In Tuffy, we opted for the "Diagonal Newton" algorithm. The Newton method can converge to the minimum in a single step. To do this, it multiplies by the inverse gradient of the Hessian matrix [17]. In RockIt no weight learning had been implemented yet when we developed ArThUR. For the Inference Process, we have chosen the "Marginal inference" method and MC-SAT algorithm because it appears that it is the most effective method in [18]. The fundamental difference between the MC-SAT algorithm and other sampling algorithms is that MC-SAT provides a certain robustness. Hence, we have chosen this algorithm for both Tuffy and Alchemy. On the other hand, RockIt implements a Gibbs sampling algorithm. This is a method where at each iteration the algorithm selects any atom and

varies its truth value. If after variation there is no strong constraints, cardinality constraints or existential constraints are violated then the probability of the atom is updated.

Once the methods has been selected, we used our tool to obtain the runtime of three MLR for learning and inference, and to get the results returned by these three reasoners. In this context, we have selected various ontologies on the internet and from the Orthogen Project [4] that conform to the OWL2 standard. The ontologies used from Orthogen Project range from 2 to 621 rules, from 50 to 531 declarations and from 40 to 383 evidence or individuals. Figures 3 and 4 below, show the runtime according to the number of rules, this factor alone does not establish a new information from which we would be able to make specific predictions about our ontologies.

Finally, after a series of tests, we have opted for a relationship between both number of rules and number of ground clauses. The reason for this choice is justified by the fact that the number of rules influences the grounding phase producing ground clauses while the ground clauses have an impact on the runtime of the inference (see Figure 5 and 6).

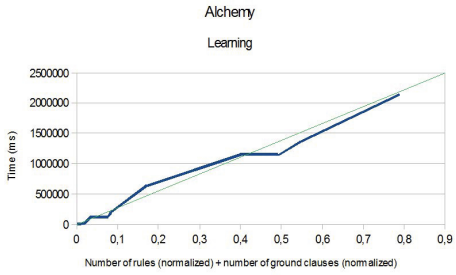

Fig. 5. Alchemy Learning

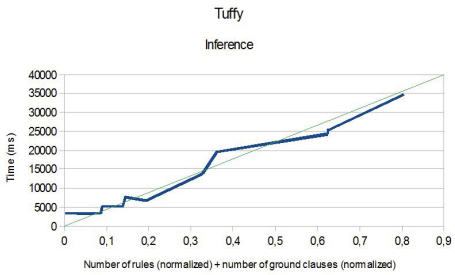

Fig. 6. Tuffy Inference

7 Conclusions

In this paper, we presented ArThUR, a tool to deal with uncertainty reasoning in SW using MLN. ArThUR presents a GUI for three reasoners : Alchemy, Tuffy and Rockit. We have shown how ArThUR can be used in practice to perform reasoning with ontologies and MLN files. The user can translate from ontologies files to MLN files, verify MLN rules and learn/infer the MLN files from the three reasoners. Moreover, we developed a statistics component and comparison tools through an XML file allowing us to share data persistence when closing the application. Secondly, we have a very fast system with a simple structure that allows search and specific queries. Moreover, from our experimental results, we conclude that the algorithms are sound, reliable and can successfully be applied to real-world domains. Finally, as future work, we want to obtain ontologies with enough facts, so that weight and rules can be learned with a high degree of confidence.

References

1. Baader, F., Calvanese, D., McGuinness, D.L., Nardi, D., Patel-Schneider, P.F.: The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press, New York (2003)
2. Berners-Lee, T., Hendler, J., Lassila, O.: The semantic web. Scientific American 284(5), 34–43 (2001)
3. Besag, J.: Statistical analysis of non-lattice data. Journal of the Royal Statistical Society. Series D (The Statistician) 24(3), 179–195 (1975)
4. DeNizza, D., Ortiz, J.J., Meurisse, H., Schobbens, P.-Y.: Orthogen: Système d'information intégré pour la traçabilité et la gestion multi-paramètres des infections orthopédiques. In: INFORSID, pp. 421–436 (2013)
5. Ding, Z., Peng, Y., Pan, R.: BayesOWL: Uncertainty Modeling in Semantic Web Ontologies. Soft Computing in Ontologies and Semantic Web, 3–29 (2006)
6. Domingos, P., Lowd, D., Kok, S., Poon, H., Richardson, M., Singla, P.: Just Add Weights: Markov Logic for the Semantic Web. In: Costa, P.C., D'Amato, C., Fanizzi, N., Laskey, K.B., Laskey, K.J., Lukasiewicz, T., Nickles, M., Pool, M. (eds.) URSW 2005-2007. LNCS (LNAI), vol. 5327, pp. 1–25. Springer, Heidelberg (2008)
7. Gruber, T.R.: A translation approach to portable ontology specifications. Knowl. Acquis. 5(2), 199–220 (1993)
8. Gu, T., Pung, H.K., Zhang, D.Q.: A bayesian approach for dealing with uncertain contexts. In: Proceedings of the Second International Conference on Pervasive Computing (2004)
9. Horrocks, I., Patel-Schneider, P.F., van Harmelen, F.: From shiq and rdf to owl: the making of a web ontology language. J. Web Sem. 1(1), 7–26 (2003)
10. Klinov, P., Parsia, B.: Probabilistic modeling and owl: A user oriented introduction to p-shiq(d). In: Dolbear, C., Ruttenberg, A., Sattler, U. (eds.) CEUR Workshop Proceedings. CEUR-WS.org, vol. 432 (2008)
11. Klir, G.J., Yuan, B.: Fuzzy Sets and Fuzzy Logic: Theory and Applications. Prentice-Hall, Inc., Upper Saddle River (1995)
12. Kok, S., Domingos, P.: Learning Markov Logic Network Structure via Hypergraph Lifting. In: Danyluk, A.P., Bottou, L., Littman, M.L. (eds.) ACM International Conference Proceeding Series, ICML, vol. 382, p. 64. ACM (2009)
13. Lukasiewicz, T.: Expressive probabilistic description logics. Artificial Intelligence 172(6-7), 852–883 (2008)
14. McGuinness, D.L., van Harmelen, F.: Owl web ontology language overview. Technical report (2004)
15. Näth, T.H., Möller, R.: Contrabovemrufum: A system for probabilistic lexicographic entailment. In: Baader, F., Lutz, C., Motik, B. (eds.) Description Logics, vol. 353 (2008)
16. Neches, R., Fikes, R., Finin, T., Gruber, T., Patil, R., Senator, T., Swartout, W.R.: Enabling technology for knowledge sharing. AI Mag. 12(3), 36–56 (1991)
17. Poon, H., Domingos, P.: Sound and efficient inference with probabilistic and deterministic dependencies. In: Proceedings of the 21st National Conference on Artificial Intelligence, AAAI 2006, vol. 1, pp. 458–463. AAAI Press (2006)
18. Singla, P., Domingos, P.: Discriminative training of markov logic networks. In: Proceedings of the 20th National Conference on Artificial Intelligence, AAAI 2005, vol. 2, pp. 868–873. AAAI Press (2005)
19. Yang, Y., Calmet, J.: Ontobayes: An ontology-driven uncertainty model. In: Proceedings of the International Conference on Intelligent Agents, Web Technologies and Internet Commerce (IAWTIC), Vienna, Austria (2005)